Sub A1

I claim:

1. A software development system for applications that run on a data network which couples a server computer and a client computer, wherein the client computer runs a browser program, comprising

a page generator capable of generating functional application pages with additional editing features for interpretation by the browser program;

an editor capable of directly operating on the pages displayed by the browser thereby allowing the user to work on a functional application during development.

2. A software development system as claimed in claim 1, further comprising a plurality of components, and wherein developed applications comprise at least one page template capable of containing components, and wherein the editor provides features to insert, modify and delete components on page templates.

3. A software development system as claimed in claim 2, wherein at least one of the components reacts interactively on user input by executing instructions on the server.

4. A software development system as in claim 3, wherein at least one of the components contains at least one other component.

5. A software development system as in claim 3, wherein the set of components displayed on pages generated from a single page template can vary for different requests of the same page template.

6. A software development system for use in a data network which couples a server computer to a client computer, wherein the client computer includes a first software program for generating a request for one or more pages from the server computer and for displaying pages, and wherein the server computer includes a second software program for receiving and processing the request from the client computer, for generating and storing pages, and for

transmitting pages to the client computer in response to requests, the server computer further comprising:

a data store,

a plurality of components residing in the data store, including interactive components organized into component classes;

a plurality of page templates residing in the data store, at least one page template having at least one selected component incorporated therein; and

a server processor controlled by a third software program, said program providing instructions for selecting a page template based on the request from the client computer and instructions for generating a page from the page template for transmission to the client computer.

7. The development system of claim 6, further comprising a component editor controlled by a fourth software program, said program providing instructions for interactively editing selected components on a page template.

8. The development system of claim 6, wherein a component is nested within a component.

9. A method for generating documents for display by a browser using interactive components, comprising:

assigning a unique identifier to at least one of the interactive components; and

embedding the unique identifier into a generated page.

10. The method of claim 9, further comprising storing data on a server for at least one of the interactive components.

11. The method of claim 10, further comprising:

analyzing a request sent by the browser for unique identifiers; and

calling a function for the interactive components referenced by at least one of the unique identifiers contained in the request.

12. The method of claim 11, wherein at least one of the components is contained on a page template.

13. The method of claim 11, wherein at least one of the components is called by a program.

14. The method of claim 11, wherein at least one of the components is called by a another component.

15. The method of claim 11, wherein the data is stored into an object of an object oriented programming language and wherein the function is a method of the object.

16. A method for implementing client server applications, comprising:

storing data objects on a server and assigning a unique identifier to each data object;

dynamically generating a document with the unique identifier embedded in the document; and

analyzing requests for unique identifiers and calling at least one function for a data object associated with one of the unique identifiers found in the request.

17. The method of claim 16, wherein the unique identifier is embedded inside a uniform resource locator contained in a tag of the document.

18. The method of claim 16, wherein the unique identifier is embedded in scripts contained in the document.

19. The method of claim 16, wherein the unique identifier is unique within a single session.

20. The method of claim 16, wherein the unique identifier is unique within all pages generated by a single server within a defined time period.

21. The method of claim 16, wherein the data objects are created by an object-oriented programming language and said function is a method of one of these objects.

22. A server computer running an application on a data network to develop and maintain applications using a web browser, comprising:

an editor operable within the web browser for inserting, deleting, and modifying components on document templates; and

a page generator for processing document templates and for generating documents from the document templates that are understandable by the web browser.

23. A server computer as in claim 22, wherein the editor operates functional applications in an edit mode permitting editing directly in the web browser.

24. A server computer as in claim 23 wherein at least one of the components can react on user responses by executing some instructions on the server.

25. A server computer as in claim 24, wherein the server computer further comprises

a store of component classes, each component class implementing one component kind; and

a parser able to detect components marked on page templates;

wherein the editor is capable of showing a menu of components for insertion into the page templates.

26. A system to modify documents on a server in a data network which couples said server computer to a client computer, the server computer comprising:

a document store;

a first software program including instructions for transforming a first document retrieved from the document store into a second document having features which permit editing of the first document such that at least a part of the second document appears and functions similar to the first document; and

a second software program including instructions to receive information from the client computer and instructions to modify documents stored in the document store.

27. The system of claim 26, wherein the second document includes at least one component and at least one handle to indicate the position of the component to the user.

28. The system of claim 27, wherein the second document includes handles and choosing one of the handles selects an editing operation

29. The system of claim 28, wherein at least one handle indicates the position of at least one component contained in the first document and said editing operation is chosen from the group of modifying the component, deleting the component, displaying information regarding the component, and inserting a new component.

30. The system of claim 26, wherein the features are scripts.

31. The system of claim 26, wherein the scripts are generated specifically for the second document and encapsulate information which is incorporated into the first document.

32. The system as in claim 26, wherein the features incorporate information regarding the first document into the second document.

33. A system as in claim 32, wherein the information incorporated into the first document is used on the client computer in order to send change requests for the second document to the server.

34. A method for generating a page from a page template containing components, comprising:

for each component, identifying a component class of the associated component; and

creating or reusing an object of the component class.

35. The method of claim 34, further comprising calling a method of said component class to generate browser code; said method being the constructor.

36. The method of claim 34, further comprising, for all components having a name attribute, looking up the component object in session memory based on said name attribute.

37. The method of claim 34, further comprising, for all interactive components:

generating a unique identifier;

assigning said unique identifier to said object, and

embedding said unique identifier into the browser code.

38. The method of claim 37, further comprising:

inserting objects for the interactive components into a list of listening components;

working through all objects stored in the list of listening components whose bid occurs inside a name in the form data set; and

calling a method of at least one of these objects.

39. The method of claim 34, wherein the page template is parsed into a list of nodes, including text and component nodes, said method further comprising:

determining if the current node is text or a component;

if component, then calling a method for the component, comprising:

evaluating the attributes of the component if necessary;

identifying the component class associated with the component; and

calling the constructor method of the component class,

said constructor method generating browser code;

if text, then generating the text; and

repeating these steps for each node.

40. The method of claim 39, wherein at least one component contains nested components and the method of claim 39 is recursively performed for all nodes nested inside the component.